

(NASA TM 451239)

27p

STUDY OF THE PROBLEM OF SECONDARY ASSIGNMENT IN THE DESIGN

OF SEQUENTIAL SWITCHING CIRCUITS

By Jerry L. Friedman

226 refs
[1963] 64-10781
N65-89000
Code 2A

6071505

Lewis Research Center,
National Aeronautics and Space Administration
Cleveland, Ohio

ABSTRACT

10781 submitted for publication

This paper presents a method of finding upper and lower limits on the number of components used in a realization of a state table, given the number of states. Extensive use is made of Karnaugh maps in this process.

When two-level diode logic is used for the realizations of the state tables, it is found that the number of diodes required for k states, $2^{n+1} > k > 2^n$, is approximately equal to the number of diodes required for 2^{n+1} states. This is true for both the lower limit (optimum case) and the upper limit (worst case) on the number of components.

Author

INTRODUCTION

This study is concerned with the problem of binary variable assignment in the design of sequential switching circuits. Such an assignment is a necessary step in a design procedure, since it provides a basis for physical realization of a given state transition table.

This paper deals with finding upper and lower limits on the number of components used in a realization of a state table, given the number of states. These limits enclose a range in which any realization of a state table with the same number of states must fall.

The reasons for investigating this problem are numerous. Different binary variable assignments lead to different physical realizations, some of which have few components and some of which have many. Since the cost of realization of a circuit is highly dependent upon the number of components required, it is desirable to be able to design circuits which contain as few components as possible. The lower bound on the number of components required in a realization is an indication of the minimum cost and of the optimum binary variable assignment. The upper bound on the number of components is an indication of the maximum cost of realization and of the worst binary variable assignment.

The only method known to the author of assigning binary variables in a state table is trial and error. However, since there are $\frac{(2^s - 1)!}{(2^s - r)!s!}$ distinct possible assignments, where r is the number of states and s is the smallest integer providing at least $\log_2 r$ distinct state variables, this method consumes far too much time in finding a good assignment.

One aid in eliminating the bad assignments is the partitioning of state tables or connection matrices. This provides good assignments, but the optimum assignment can only be found by trial.

Thus, a method is needed to allow a sequential circuit designer to obtain directly the optimum binary variable assignment and the best possible state table by using this assignment.

NOMENCLATURE

A list of terminology used in this paper is provided here for reference purposes:

T type flip-flop = A flip-flop whose input terminal is the complementary terminal.

RS type flip-flop = A flip-flop whose input terminals are the set and reset terminals.

T_i map = A Karnaugh map which describes the switching function of the T terminal of the T type flip-flop i.

R_j map = A Karnaugh map which describes the switching function of the R terminal of the RS type flip-flop j.

S_j map = A Karnaugh map which describes the switching function of the S terminal of the RS type flip-flop j.

ϕ map = A Karnaugh map which describes the switching function of the output.

PS-NS map = An R, S, or T map.

PS = Present state.

NS = Next state.

ϕ = Output.

OC = Output class.

NC = Next class.

Acceptable entries = Entries in a state table which will not allow the number of states in the table to be reduced by any simplification process.

Compatible maps = Maps which provide acceptable entries to a state table.

Minimum map = A Karnaugh map which describes a switching function having one literal, i. e.,

1	1	1	1
1	1	1	1

T=1

1	1	0	0
1	1	0	0

T=x

0	0	0	0
0	0	0	0

T=0

Minimum function = The switching function described by a minimum map.

Maximum map = A Karnaugh map which has alternating "1's" and "0's" both vertically and horizontally, i. e.,

1	0	1	0
0	1	0	1

0	1	0	1
1	0	1	0

Maximum function = The switching function described by a maximum map.

Optimum case maps or optimum set of maps = A set of compatible PS-NS

and ϕ maps which describe a set of switching functions, the realization of which contains the fewest total number of diodes possible when two-level diode logic is used.

Optimum case functions or optimum set of functions = The set of switching functions determined by the optimum set of maps.

Optimum case = Optimum set of maps and the optimum set of functions.

Worst case maps or worst set of maps = A set of compatible PS-NS and ϕ maps which describe a set of switching functions, the realization of which contains the largest total number of diodes possible when two-level diode logic is used.

Worst case functions or worst set of functions = The set of switching functions determined by the worst set of maps.

Worst case = Worst set of maps and the worst set of functions.

T_{\max} = Worst case when T type flip-flops are used.

T_{\min} = Optimum case when T type flip-flops are used.

RS_{\max} = Worst case when RS type flip-flops are used.

RS_{\min} = Optimum case when RS type flip-flops are used.

Optimum and Worst State Tables

The problem presented in the introduction is stated here in more detail. Given the number of states, find the state table which has a set of switching functions describing the memory elements with the smallest possible number of literals. This table and its switching functions are called optimum. The circuit realization of this optimum case contains the smallest number of components possible. Also, find the state table which has a set of switching functions describing the memory elements with the largest possible number of literals. This is the worst case and requires the maximum number of components in the realization. Thus, the upper and the lower limits on the number of components required in a realization are determined by the optimum and the worst cases.

In the search for an optimum set and a worst set of functions and the corresponding state tables for a specified number of states, it is necessary to consider only non-trivial cases. A non-trivial case is one in which the number of states cannot be reduced by any simplification techniques.

Both RS type and T type flip-flops are considered as memory elements, and there is only a single output line.

a) The Approach to the Optimum Case

The approach to the optimum case will now be described. A single approach is used to cover both the RS_{\min} and the T_{\min} cases. Consider a group of states, where the number of states ranges between $2^n + 1$ and 2^{n+1} , because within this group, the RS_{\min} maps are similar to each other and

the RS_{min} state tables are similar to each other. The same is true for T_{min} .

The first step is to write a set of minimum PS-NS Karnaugh maps for the number of states equal to 2^{n+1} . A minimum map is a map which yields a minimum switching function, that is, a switching function such as $F=0$, $F=1$, $F=X$, or any switching function with only one literal. Any pattern of ones and zeros on the map is permissible as long as the switching function is minimum. A minimum map should be drawn to describe each flip-flop input. For the number of states between $2^n + 1$ and 2^{n+1} , there are $n + 1$ flip-flops and therefore $n + 1$ inputs described by a set of $n + 1$ maps.

A state table should then be drawn from the set of PS-NS maps. Next, an attempt should be made to draw a minimum map for the output which will provide acceptable outputs in the state table. Acceptable outputs are defined as those outputs which can be entered into the state table without causing a trivial case to occur. In other words, the number of states in the resulting state table should not be able to be reduced by any simplification techniques.

If this minimum output map will not provide acceptable outputs, then a different minimum map should be drawn, and the outputs should be entered into the state table and checked for acceptability.

If such an output map is found, then switching functions can be written from the PS-NS and ϕ maps, and the total number of components used in the realization of these functions can be computed. This total number is obtained by summing the number of components needed in the

realization of each input of each flip-flop and the output for a specific number of states. If the PS-NS and ϕ maps are minimum maps, the resulting optimum switching functions will be minimum.

If an acceptable minimum output map is not found, then draw any output map whose switching function has two literals. If these output entries are acceptable in the state table, write the switching functions that describe the set of PS-NS and ϕ maps. If the entries are not acceptable, draw different output maps that have switching functions of two literals, and check for acceptability in the state table. If no acceptable map with a switching function of two literals is found, draw a map whose switching function has three literals. Continue this process until an acceptable output map can be found. Then write the switching functions from the PS-NS and ϕ maps, and calculate the number of components required in the realization.

In some cases, all the PS-NS maps in a set cannot be minimum for an acceptable output map to exist. An example of such a case would be one in which the resulting state table from a set of minimum PS-NS maps looks as follows:

Present State (PS)	Next State (NS)	
	0	1
A	A	A
B	A	A
C	A	A
D	A	A
E	A	A
F	A	A

Since there are only four possible output combinations (four output classes) which can be written, i.e.,

Output		Output Class
0	1	
0	0	W
0	1	X
1	0	Y
1	1	Z

then only four of the next state pairs (NS 0,1) can be identical, and a different output class can be associated with each. If more than four next state pairs are identical, then the number of states can be reduced by a simplification process.

If a set of minimum PS-NS maps yields a state table for which an acceptable output map cannot be found, then one or more of the PS-NS maps must be altered until an acceptable output map can be found.

The procedure for doing this is exactly the same as the one for finding an acceptable output map. Alter one PS-NS map in order to give a different minimum map. If changing one of the PS-NS maps from one minimum combination to another does not solve the problem, the same procedure should be tried with more than one of the PS-NS maps.

If it is not possible to find a set of minimum PS-NS maps for which there exists an acceptable output map, then alter one of the PS-NS maps so that its switching function contains two literals. Again try to find an acceptable output map. If this is still impossible, alter the PS-NS map so that the number of literals in its switching function increases by one again; continue this process until an acceptable output map can be

found. Then write the appropriate switching functions, and calculate the number of components required in the realization.

If all the PS-NS and ϕ maps are not minimum, check to make sure there are no alterations of any of the maps which will give switching functions closer to the minimum. The first trial for obtaining the optimum case has now been completed for 2^{n+1} states.

The next problem is to find the optimum case for $2^{n+1} - 1$ states. In this case, there is one don't-care condition. If it is possible, choose as the don't-care condition one of the present states which is not a next state. In other words, in the example below, C is not one of the next states. Therefore, choose C as the don't-care condition for the three state case.

Present State	Next State	
	0	1
A	A	A
B	B	B
C	A	A
D	A	A

If this is done, the set of maps for $2^{n+1} - 1$ states will yield switching functions which are either a minimum or closer to the minimum than for 2^{n+1} states. The reason for this is that the maps for the $2^{n+1} - 1$ state case will be exactly the same as for the 2^{n+1} state case, except that there will be one don't-care condition which might improve, but can never hurt, the map patterns. The procedure to be carried out is identical to that for 2^{n+1} states.

Next, the problem is to find the optimum case for $2^{n+1} - 2$ states. Again, choose as don't-care conditions present states which are not next

states, and solve as for $2^{n+1} - 1$ states. In a similar manner, solve for $2^{n+1} - 3$, $2^{n+1} - 4$, ..., $2^n + 1$ states.

Often, the solution for $2^n + 1$ states provides information leading to better solutions (closer to the minimum) for $2^n + 2$, $2^n + 3$, ..., 2^{n+1} states. Therefore, once the solution for $2^n + 1$ states has been found, then this form of solution (the same map patterns) should be tried with $2^n + 2$, $2^n + 3$, ..., 2^{n+1} states. In all cases, the solutions obtained at the end of this procedure are optimum.

Fig. 1 shows this process in the form of a flow chart.

b) The Approach to the Worst Case

The approach to the worst case will now be described. A single approach is used to cover both the RS_{max} and the T_{max} cases. As in the optimum case, consider a group of states, where the number of states ranges between $2^n + 1$ and 2^{n+1} . The reason for doing this is that the RS_{max} maps are similar to each other, as are the RS_{max} state tables. The same is true for T_{max} .

The first step is to draw a set of maximum PS-NS Karnaugh maps for 2^{n+1} states. A maximum map alternates between ones and zeros horizontally and vertically. One maximum map should be drawn to describe each of the $n + 1$ input terminals to the $n + 1$ flip-flops. A state table should be drawn from the set of $n + 1$ PS-NS maps, and an attempt should then be made to draw a maximum output map which will provide acceptable outputs in the state table. If this output map will not provide acceptable outputs, then a different maximum output map should be drawn, and its outputs should be entered into the state table to see if they are acceptable.

At this point it should be noted that there are only two different maximum maps which can be drawn. One has a "1" in the 0, 0, ..., 0 position, while the other has a "0" in this position. The remainder of the entries alternate between "0" and "1".

If an acceptable output map is found, then switching functions can be written from the PS-NS and ϕ maps, and the number of components used in the realization can be computed. Provided the PS-NS and ϕ maps are maximum, the resulting worst case switching functions will be maximum.

If neither of the maximum output maps is acceptable, then a map with a switching function of one less literal than the maximum case should be drawn and tested for acceptability. If no output map with this number of literals is acceptable, draw a map with a switching function of one less literal than the previous one and check its entries in the state table. Continue this process until an acceptable output map is found. Then write the switching functions describing the PS-NS and ϕ maps, and calculate the number of components used in the realization.

In some cases, all the PS-NS maps in a set cannot be maximum for there to exist an acceptable output map. The same example can be cited as in the section on optimum functions. This case arises when more than four of the next states have the same pairs of entries in the state table. If such a case arises, then one or more of the PS-NS maps must be altered until an acceptable output map can be found.

The procedure for doing this is exactly the same as that for finding an acceptable output map. Alter one PS-NS map in order to give a different maximum map. If this does not solve the problem, alter more than one

PS-NS map to give different maps, still maximum. If it is impossible to find a set of maximum PS-NS maps for which there exists an acceptable output map, then alter one of the PS-NS maps so that its switching function is less than the maximum function by one literal. If this still does not provide a solution, alter the map so that the switching function is reduced from the maximum by two, three, or more literals until an acceptable output map can be found. Then, write the switching functions, and compute the number of components used in the realization.

If all the PS-NS and ϕ maps are not maximum, check to make sure there are no alterations of any of the maps which will give switching functions closer to the maximum. The first trial for obtaining the worst case has now been completed for 2^{n+1} states.

The next problem is to find the worst case for $2^{n+1} - 1$ states. In this case, there is one don't-care condition. If it is possible, choose as the don't-care condition one of the present states which is not a next state, or choose a state which will cause the output map to be improved. If the number of don't-care conditions is even, choose as the don't-care conditions states which go to each other as next state pairs, if possible.

In other words, if there are two don't-care conditions, choose either A and F, or B and E, or D and C in the four state case.

Present State	Next State	
	0	1
A	F	A
B	B	E
C	D	C
D	D	C
E	B	E
F	F	A

After the don't-care conditions are chosen, proceed as in 2^{n+1} states.

The next step is to find the worst case for $2^{n+1} - 2$ states. Proceed as for $2^{n+1} - 1$ states. Similarly solve the cases for $2^{n+1} - 3$, $2^{n+1} - 4$, ..., $2^n + 1$ states.

Often, the solution for $2^n + 1$ states provides information leading to better solutions (closer to the maximum) for $2^n + 2$, $2^n + 3$, ..., 2^{n+1} states. Therefore, once the solution for $2^n + 1$ states has been found, then this form of solution (the same map patterns) should be tried with $2^n + 2$, $2^n + 3$, ..., 2^{n+1} states. In all cases, the solutions obtained at the end of this procedure are the worst cases.

Fig. 2 shows this process in the form of a flow chart.

Figs. 3 and 4 and table I show the number of diodes required in optimum case and worst case realizations that use two-level diode logic.

CONCLUSIONS

The method developed in this paper for finding the optimum and the worst state tables for a given number of states is one of trial and error. However, the procedure followed is quite systematic and allows a designer to obtain the optimum or the worst state table in only a few trials, since there are only a finite number of distinctly different state tables resulting from the finite number of given states. Furthermore, there are only a small number of distinct and different state tables which even approach the optimum and the worst cases. Therefore, relatively few trials are required in order to find the desired tables.

There are no restrictions on the number of states for which this approach is applicable. However, the use of Karnaugh maps becomes cumbersome when there are more than 64 states. Therefore, since Karnaugh

maps are used in this approach, an upper limit of 64 states might be imposed by the designer.

This approach is geared to realization by two-level diode logic, since, in this logic mode, the number of components in a realization is directly proportional to the number of literals in the switching function. This property is assumed to hold in this approach.

Figs. 3 and 4 showing the range of the number of diodes required for realization in two-level logic of a state table with a given number of states, allow the drawing of several conclusions. First, the number of diodes required for k states, where $2^{n+1} > k > 2^n$, is approximately equal to the number of diodes required for 2^{n+1} states. This is true for both the optimum and the worst cases and the T type and the RS type flip-flops. If $D(y)$ is the number of diodes required for y states and n is the number of flip-flops, then

$$D(k) \approx D(2^{n+1})$$

where $k > 2^n$. Second, it is not necessary to compute the maximum or the minimum number of diodes for each number of states, since the range is well established by only a few states between 2^{n+1} and $2^n + 1$.

The similarity between state tables of any one type (e.g., RS_{max}) containing between 2^{n+1} and $2^n + 1$ states can also be seen.

Biographical Sketch

Jerry L. Friedman was born in Brooklyn, New York, on July 18, 1939. He received the B.E.E. degree from Cornell University, Ithaca, New York, in June 1962 and the M.S. degree in Electrical Engineering from Cornell University in June 1963.

Preferred address for correspondence and return of proofs is:
77 Stanton Road, Darien, Connecticut, 06823

Since June 1963, he has been an Aerospace Systems Engineer at the National Aeronautics and Space Administration Lewis Research Center in Cleveland, Ohio, where he is involved in Electromagnetic Interference and Compatibility Problems.

Mr. Friedman is a member of Eta Kappa Nu, the Cornell Society of Engineers, and the IEEE.

REFERENCES

1. D. Aufenkamp, "Analysis of Sequential Machines II," IRE Trans. on Elect. Computers, EC-7, pp. 299-306, Dec. 1958.
2. D. Aufenkamp, and F. Hohn, "Analysis of Sequential Machines I," IRE Trans. on Elect. Computers, EC-6, pp. 276-285, Dec. 1957.
3. S. H. Caldwell, Switching Circuits and Logical Design, Wiley, N.Y., 1958.
4. A. Gill, "Characterizing Experiments for Finite-Memory Binary Automata," IRE Trans. on Elect. Computers, EC-9, pp. 469-471, Dec. 1960.
5. S. Ginsburg, "A Synthesis Technique for Minimal State Sequential Machines," IRE Trans. on Elect. Computers, EC-8, pp. 13-25, Mar. 1959.
6. S. Ginsburg, "A Technique for the Reduction of a Given Machine to a Minimal-State Machine," IRE Trans. on Elect. Computers, EC-8, pp. 346-356, Sept. 1959.
7. S. Ginsburg, "Synthesis of Minimal-State Machines," IRE Trans. on Elect. Computers, EC-8, pp. 441-449, Dec. 1959.
8. W. S. Humphrey, Switching Circuits, McGraw-Hill, N.Y., c. 1958.
9. R. S. Ledley, Digital Computer and Control Engineering, McGraw-Hill, N.Y., c. 1960.
10. E. McCluskey, and S. Unger, "A Note on the Number of Internal Variable Assignments for Sequential Switching Circuits," IRE Trans. on Elect. Computers, EC-8, pp. 439-441, Dec. 1959.

11. R. McNaughton, and H. Yamada, "Regular Expressions and State Graphs for Automata," IRE Trans. on Elect. Computers, EC-9, pp. 39-48, Mar. 1960.
12. G. Mealy, "A Method for Synthesizing Sequential Circuits," Bell System Tech. Jour., vol. 34, pp. 1045-1079, Sept. 1955.
13. Mezei, "Minimal Characterizing Experiments for Finite Memory Automata," IRE Trans. on Elect. Computers, EC-10, p. 288, June 1961.
14. M. Phister, Logical Design of Digital Computers, Wiley, N.Y., Sixth Printing, c. 1958.
15. J. Simon, "A Note on Memory Aspects of Sequence Transducers," IRE Trans. on Circuit Theory, CT-6, pp. 26-29, Mar. 1959.

Table I - Number of diodes required in realizations
using two-level diode logic

Number of states	T _{min}	T _{max}	RS _{min}	RS _{max}	Number of flip-flops
1	0	8	0	14	1
2	0	12	0	18	1
3	0	42	0	77	2
4	0	42	0	73	2
5	4	151	4	266	3
6	4	152	4	264	3
7	4	150*	6	260*	3
8	6	147	6	256	3
9	6	451	6	826	4
10	6	449	6	820	4
11	6	445*	6	815*	4
12	6	442	6	808	4
13	6	433	6	797	4
14	6	430*	6	787*	4
15	9	428	6	779	4
16	10	434	6	771	4

* Predicted from figs. 3 and 4.

Figure Legend

Fig. 1 - Flow chart for optimum case.

Fig. 2 - Flow chart for worst case.

Fig. 3 - The range (minimum to maximum) of the number of diodes required in the realization of a map, using T-type flip-flops and two-level diode logic, given the number of states.

Fig. 4 - The range (minimum to maximum) of the number of diodes required in the realization of a map, using RS-type flip-flops and two-level diode logic, given the number states.

Table I - Number of diodes required in realizations using two-level diode logic.

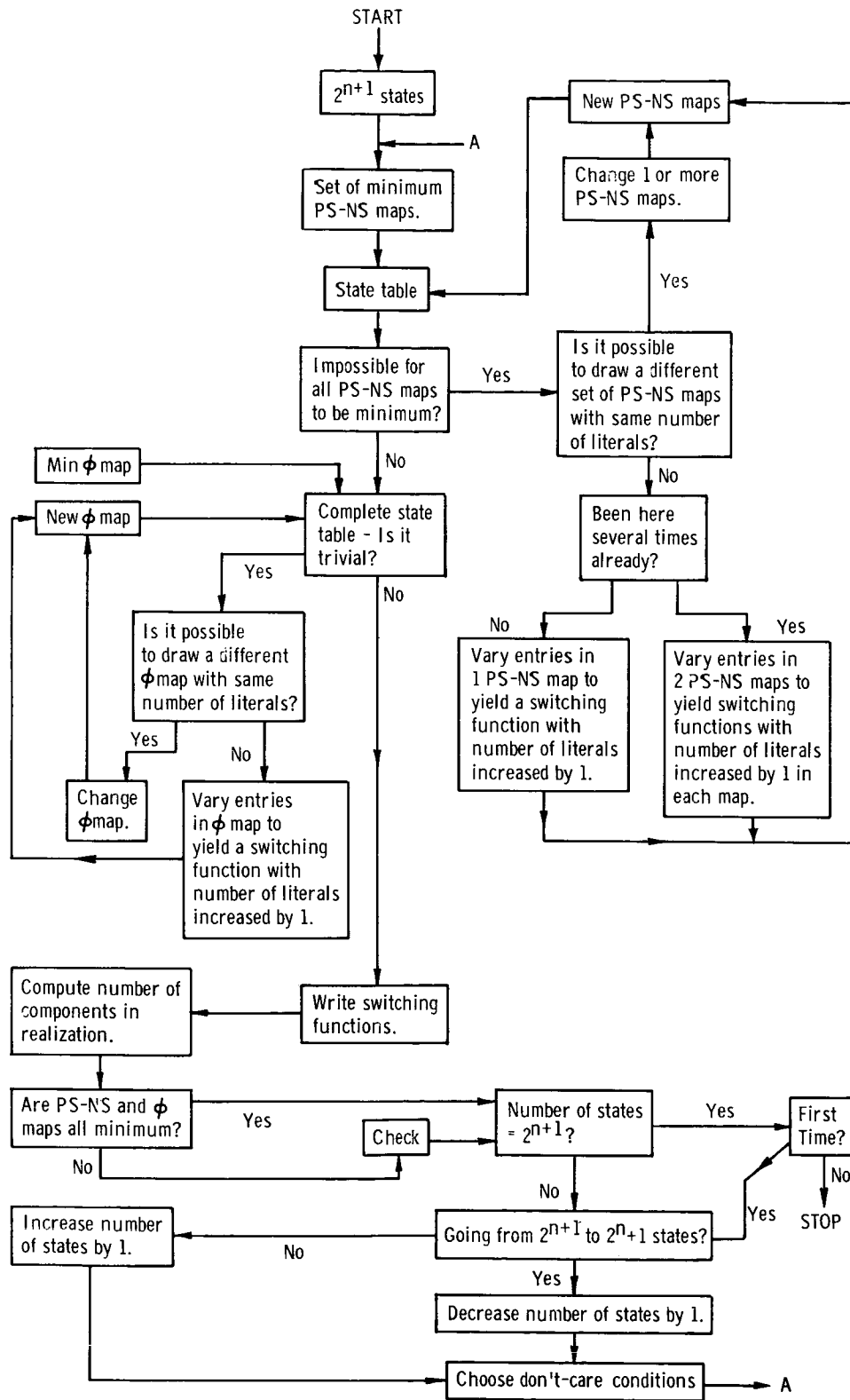


Fig. 1. - Flow chart for optimum case.

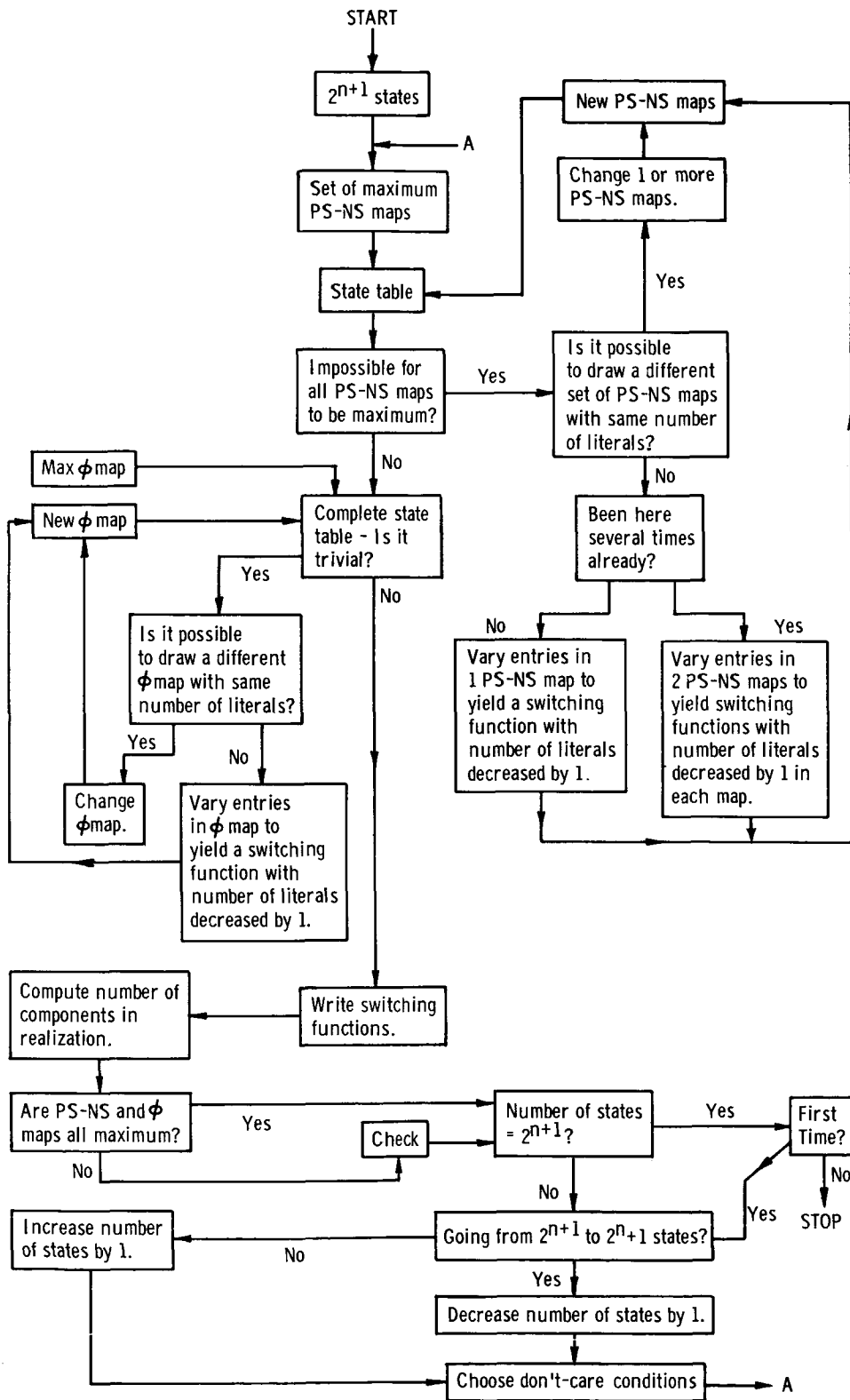


Fig. 2. - Flow chart for worst case.

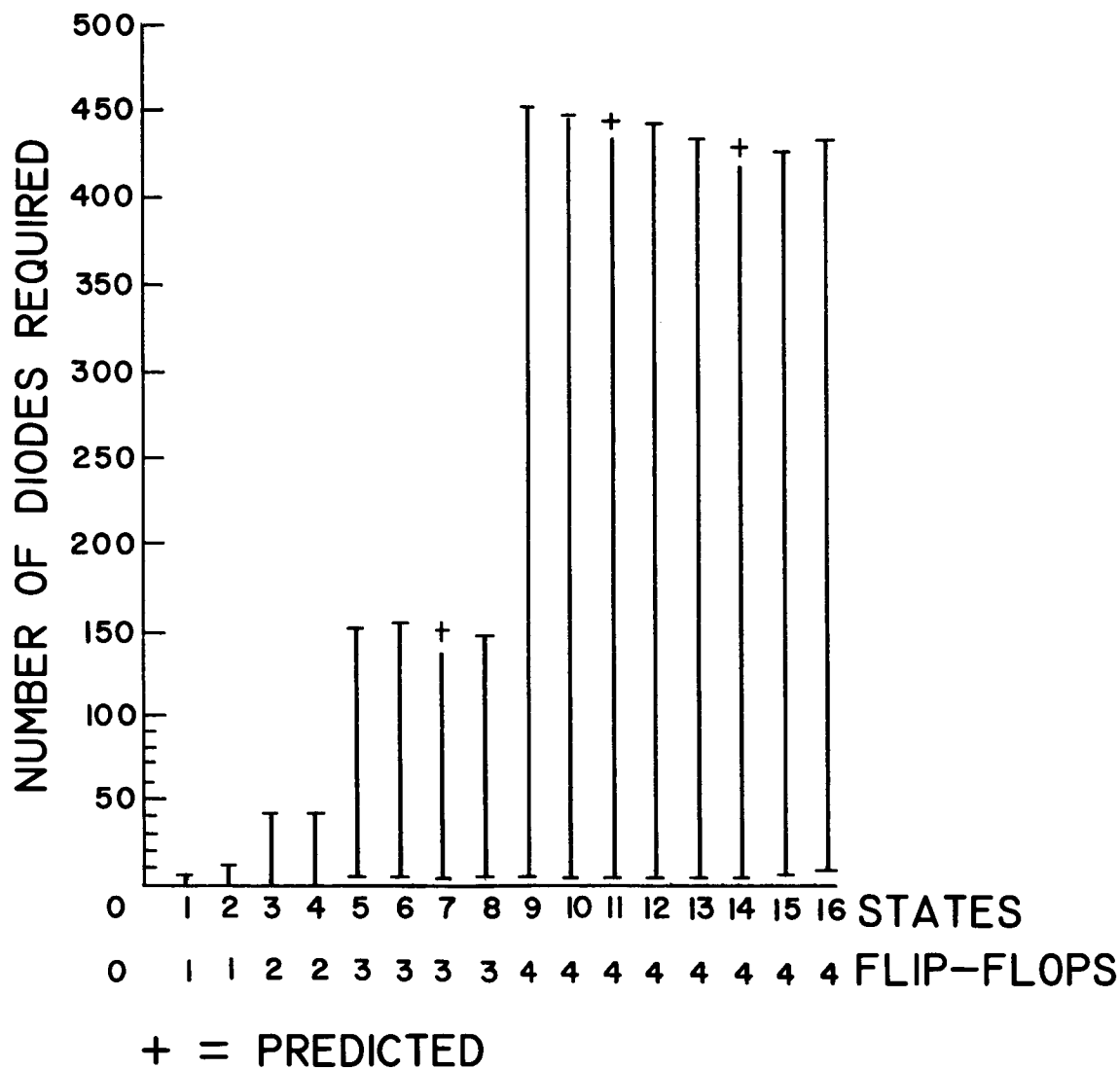


Fig. 3. - The range (minimum to maximum of the number of diodes required in the realization of a map, using T type flip-flops and two-level diode logic, given the number of states.

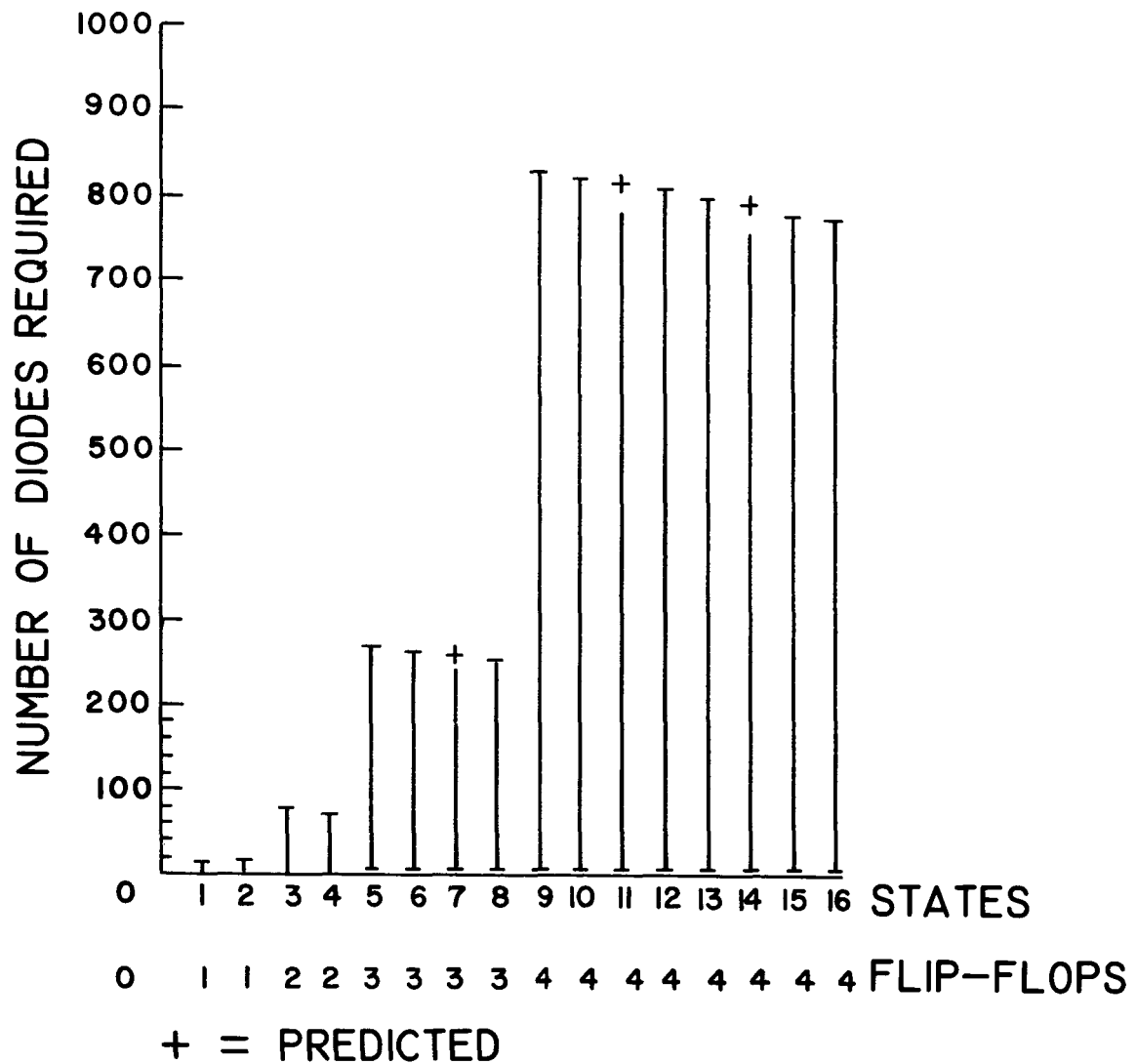


Fig. 4. - The range (minimum to maximum) of the number of diodes required in the realization of a map using RS type flip-flops and two-level diode logic, given the number of states.